Evaluación de Rendimiento y Eficiencia Energética de Sistemas Heterogéneos para Bioinformática

Enzo Rucci
Directores por UNLP: Armando De Giusti y Marcelo Naiouf
Directores por UCM: Carlos García Sanchez y Guillermo Botella Juan
Fecha de exposición: 10/03/2016
Facultad de Informática, Universidad Nacional de La Plata, Argentina
erucci@lidi.info.unlp.edu.ar

1. Introducción

En primer lugar, se presenta la motivación de esta tesis. Luego se enuncian los objetivos y la metodología a emplear. Finalmente, se describen los alcances y limitaciones de esta investigación.

1.1. Motivación

En un mundo con recursos energéticos limitados y una constante demanda por mayor poder computacional, el problema del consumo energético se presenta como uno de los mayores obstáculos para el diseño de sistemas que sean capaces de alcanzar la escala de los Exa_ops. Por lo tanto, la comunidad científica está en la búsqueda de diferentes maneras de mejorar la eficiencia energética de los sistemas de cómputo de altas prestaciones (HPC, por sus siglas en inglés) [1].

Una estrategia reciente para incrementar el poder computacional de los sistemas HPC consiste en incorporarles aceleradores y coprocesadores, como pueden ser las unidades de procesamiento gráfico (GPU, por sus siglas en inglés) de AMD y NVIDIA o los coprocesadores Xeon Phi de Intel. Más allá de que estos dispositivos no comparten las mismas características, la estrategia se basa en utilizarlos para las secciones de código que requieren cómputo intensivo y dejar las tareas livianas a la unidad central de procesamiento (CPU, por sus siglas en inglés), como la entrada/salida (E/S) y las comunicaciones. Estos sistemas híbridos que emplean diferentes recursos de procesamiento se denominan arquitecturas heterogéneas y son capaces de obtener picos de rendimiento muy superiores a los de las CPUs. En los últimos años, esta estrategia se ha ido consolidando por una capacidad adicional de los aceleradores y coprocesadores: el incremento en el poder de cómputo se logra al mismo tiempo que se limita el consumo de potencia, lo que permite obtener mejores cocientes FLOPS/Watt. El ranking TOP500 [2] sirve para dar evidencia a esta tendencia mientras que el Green500 [3] la refuerza. En noviembre de 2010, ambos rankings contaban con 17 sistemas con aceleradores. Sin embargo, cinco años más tarde, este número se incrementó a 104 para el TOP500 y a 103 para el Green500.

Uno de los primeros aceleradores en ser incorporados a los sistemas de cómputo de altas prestaciones fueron las GPUs. En sus inicios estos dispositivos fueron pensados para el procesamiento gráfico y sus diseños estuvieron orientados a este tipo de aplicaciones. En la última década, se introdujeron algunas modificaciones a las arquitecturas de estos dispositivos y se desarrollaron librerías que permiten evitar el uso de primitivas de gráficos para programarlos. Estas modificaciones posibilitaron una amplia extensión de su uso. La clave del poder computacional de las GPUs se encuentra en su arquitectura: cientos o miles de núcleos pequeños y simples que ejecutan instrucciones en orden y operan en grupo como un procesador vectorial sumado a una memoria principal con alto ancho de banda, la convierten en un arquitectura masivamente paralela capaz de lograr un pico de

cados de forma tal que reflejen su arquitectura para lograr alto rendimiento. En ese sentido, tanto la arquitectura de las GPUs como sus modelos de programación difieren sustancialmente de los correspondientes a las CPUs, que es donde la comunidad HPC se ha focalizado tradicionalmente. En consecuencia, el programador se ve obligado a aprender conocimientos específicos de las arquitecturas y de su programación, lo que representa una tarea ardua.

Una alternativa a las GPUs son los coprocesadores Xeon Phi, los cuales fueron desarrollados por Intel para aplicaciones que requieren de cómputo de altas prestaciones. Se diseñaron como un complemento de las CPUs aunque pueden operar en forma individual, lo que los diferencia de las placas gráficas. Su arquitectura se basa en la de los procesadores Xeon pero con un mayor número de núcleos y de capacidades vectoriales más amplias. Al compartir los mismos modelos de programación que los procesadores Xeon, no resulta necesario aprender un nuevo lenguaje o herramienta para programarlos, con lo cual se reduce el tiempo requerido para desarrollar o portar una aplicación en este tipo de dispositivos. Por haber sido desarrollado recientemente, su utilidad y rendimiento son evaluados en la actualidad en diferentes áreas de aplicación, como pueden ser álgebra lineal, predicción del clima o diferentes tipos de simulación [4].

Las *Field Programmable Gate Arrays (FPGAs)* son otro tipo de acelerador basado en circuitos integrados reconfigurables. Aunque tradicionalmente fueron utilizadas para el procesamiento digital de señales, su uso ha crecido en diferentes áreas y probablemente se expanda aun más considerando que Intel compró recientemente Altera, una de las principales empresas fabricantes de FPGAs [5]. HPC es una de las áreas que se ha interesado en estos aceleradores debido principalmente a la evolución en su capacidad de cómputo y a su bajo consumo energético. Si bien tanto la frecuencia del reloj como el pico de rendimiento suelen ser más bajos que los correspondientes a las CPUs y a las GPUs, la capacidad de configurar el hardware para que se adapte al problema específico a resolver le da la posibilidad a las FPGAs de obtener mejores rendimientos, además de ser en general más eficientes desde el punto de vista energético [6]. La programación de estas placas se realiza a través de lenguajes de descripción de hardware (HDL, por sus siglas en inglés), como pueden ser Verilog o VHDL. Desafortunadamente, los HDLs son engorrosos, propensos a errores y requieren tener que mantener una explícita noción del tiempo, lo que implica una complejidad adicional [7]. En la actualidad, las principales empresas fabricantes de FPGAs (Alteray Xilinx) trabajan en el desarrollo de herramientas que permitan disminuir el esfuerzo de programación de estos dispositivos; en particular, a través de OpenCL, lo que resulta más familiar para los programadores HPC.

Más allá del tipo de acelerador utilizado, la programación de sistemas heterogéneos representa un verdadero desafío. Para lograr aplicaciones de alto rendimiento, los programadores deben enfrentar una serie de dificultades como pueden ser: estudiar características específicas de cada arquitectura, contemplar aplicaciones con múltiples niveles de paralelismo y aplicar técnicas de programación y optimización particulares para cada una de ellos, lograr una distribución del trabajo y un balance de carga entre los diferentes dispositivos de procesamiento que permita obtener buen rendimiento y afrontar el surgimiento de nuevos lenguajes y modelos de programación junto a la ausencia de un estándar y de herramientas afianzadas para este tipo de sistemas.

Una de las áreas que se ve afectada por los problemas actuales de los sistemas HPC es la bioinformática, ya que cuenta con un número creciente de aplicaciones que requieren de cómputo de altas prestaciones para alcanzar tiempos de respuesta aceptables. Esta situación se debe principalmente al crecimiento exponencial que ha experimentado la información biológica en los últimos años [8].

Una operación fundamental en cualquier investigación biológica es el alineamiento de secuencias, la cual consiste en comparar dos o más secuencias biológicas, como pueden ser las de ADN o las de proteínas. El propósito de esta operación es detectar qué regiones dentro de una secuencia comparten una historia evolutiva común. El alineamiento de secuencias resulta esencial en el análisis filogenético, el perfilado de enfermedades genéticas, la identificación y cuantificación de regiones conservadas o unidades funcionales, y el perfilado y predicción de secuencias ancestrales. También resulta importante en el desarrollo de nuevas drogas y en la investigación forense criminal, y es por ello que actualmente la comunidad científica realiza un gran esfuerzo en este campo de la bioinformática [9].

El algoritmo de Smith-Waterman (SW) [10] es un método popular para el alineamiento local de secuencias que ha sido utilizado como base para otros algoritmos posteriores y como patrón con el cual comparar otras técnicas de alineamiento. Sin embargo, su complejidad computacional es cuadrática, lo que lo hace inviable para procesar grandes conjuntos de datos biológicos. En la actualidad, se emplean diversas heurísticas que permiten reducir el tiempo de ejecución a costo de una pérdida en la sensibilidad de los resultados. Por fortuna, el proceso de alineamiento exhibe cierto paralelismo inherente que puede ser explotado para reducir el costo computacional de SW sin perder precisión.

De manera de procesar el creciente volumen de información biológica con tiempos de respuesta aceptables, es necesario desarrollar nuevas herramientas computacionales que sean capaces de acelerar primitivas claves y algoritmos elementales, especialmente aquellos que son costosos computacionalmente como es el caso del algoritmo SW. En ese sentido, el empleo de paralelismo se vuelve fundamental y para ello resulta indispensable contar con una evaluación integral de los diferentes sistemas HPC que considere el rendimiento alcanzable y el consumo energético asociado.

1.2. Objetivos y metodología

El objetivo general de esta tesis consiste en evaluar el rendimiento y la eficiencia energética de sistemas para cómputo de altas prestaciones al acelerar el alineamiento de secuencias biológicas mediante el algoritmo SW.

Los objetivos específicos son los siguientes:

- Describir y comparar las propuestas científicas para el alineamiento de secuencias biológicas utilizando sistemas para cómputo de altas prestaciones con el objeto de caracterizarlas.

- Identificar los factores que inciden en el rendimiento y la eficiencia energética de los sistemas para cómputo de altas prestaciones al acelerar el alineamiento de secuencias.
- Diseñar y desarrollar soluciones algorítmicas para aquellos sistemas de cómputo de altas prestaciones que no tengan implementaciones disponibles o, en caso contrario, que superen a las existentes en cuanto a aceleración.
- Medir y analizar el rendimiento y la eficiencia energética de las herramientas seleccionadas y de las propuestas desarrolladas.

Esta investigación doctoral se plantea como un estudio proyectivo que pretende dar solución a la necesidad de contar con implementaciones aceleradas del alineamiento de secuencias mediante el algoritmo SW en sistemas de cómputo de altas prestaciones junto a evaluaciones de rendimiento que además consideren la eficiencia energética.

1.3. Alcances y limitaciones

Con respecto al caso de estudio seleccionado, se decidió focalizar en el alineamiento de proteínas en lugar del de ADN por dos razones. En primer lugar, porque resulta ventajoso desde el punto de vista biológico en la mayoría de los casos [11]. En segundo lugar, el alineamiento de secuencias de proteínas es más complejo debido a un alfabeto más grande y al uso de matrices de sustitución, por lo que resolver el alineamiento de secuencias de ADN resulta relativamente simple una vez resuelto el primero.

En relación a la explotación de sistemas heterogéneos, las GPUs son el acelerador dominante en la comunidad de HPC al día de hoy por su alto rendimiento y bajo costo de adquisición y existe vasta investigación científica sobre el uso de esta clase de acelerador para el alineamiento de secuencias como se describe en la Sección 2.2.2. Por ese motivo se decidió priorizar el desarrollo de nuevas soluciones algorítmicas para sistemas heterogéneos basados en coprocesadores Xeon Phi y basados en FPGAs. Los coprocesadores Xeon Phi fueron introducidos recientemente y, al inicio de esta tesis, no existían implementaciones disponibles para el alineamiento de secuencias. En cuanto a las FPGAs, el estudio de su aplicación para el procesamiento de secuencias biológicas sí cuenta con antecedentes aunque estas implementaciones fueron desarrolladas con HDLs tradicionales y en su mayoría poseen una o más limitaciones que restringen su uso en el mundo real (ver Sección 2.2.2). El reciente desarrollo de herramientas basadas en OpenCL por parte de las principales empresas fabricantes de FPGAs representa uno de los motivos del estudio de esta clase de acelerador para el alineamiento de secuencias.

2. Alineamiento de secuencias biológicas de alto rendimiento

Esta sección se centra en la fuerte relación entre el alineamiento de secuencias biológicas y el HPC.

2.1. Algoritmo SW para alineamiento de secuencias biológicas

El algoritmo SW computa el alineamiento local óptimo entre dos secuencias siguiendo un enfoque matricial y se puede dividir en dos fases: (1) Cómputo de la matriz de similitud (o también llamada matriz de alineamiento), para obtener el puntaje de alineamiento óptimo; y (2) Retroceso, para obtener el alineamiento óptimo.

1. Cómputo de la matriz de similitud: dadas dos secuencias q = q1q2q3. . . qm y d = d1d2d3. . . dn, se construye una matriz de similitud H de (m+1)_(n+1) elementos. Las relaciones de recurrencia para completar la matriz, con las modificaciones de Gotoh para admitir un modelo de penalización de gap por afinidad [12], se detallan a continuación:

$$H_{i,j} = max \begin{cases} 0 \\ H_{i-1,j-1} + SM(q_i, d_j) \\ E_{i,j} \\ F_{i,j} \end{cases} \qquad E_{i,j} = max \begin{cases} H_{i,j-1} - G_{oe} \\ E_{i,j-1} - G_e \end{cases}$$

$$F_{i,j} = max \begin{cases} H_{i-1,j} - G_{oe} \\ F_{i-1,j} - G_e \end{cases}$$

Los residuos de la secuencia q, usualmente llamada secuencia de consulta, etiquetan las _las mientras que los residuos de la secuencia d, usualmente llamada secuencia de la base de datos. Hi;j representa el puntaje de alineamiento para las subsecuencias de q y d terminadas en los residuos qi y dj , respectivamente. Ei;j y Fi;j son los puntajes de los alineamientos de las mismas subsecuencias de q y d pero terminadas en un gap en q y en d, respectivamente. SM es la matriz de sustitución que define los puntajes de sustitución entre todos los pares de residuos. Matrices de sustitución usuales en el alineamiento de secuencias de proteínas son las familias BLOSUM o PAM. Goe es la suma de las penalizaciones por inserción y extensión de gap, mientras que Ge es la penalización

por extensión de gap. Hi;j , Ei;j y Fi;j son inicializadas en 0 cuando i = 0 o j = 0. El puntaje de alineamiento óptimo S es el máximo puntaje de alineamiento de la matriz H.

2. Retroceso: el alineamiento óptimo se calcula realizando un proceso de retroceso desde la posición donde se encuentra el puntaje de alineamiento óptimo S hasta llegar a una posición donde el valor sea 0, siendo éste punto el inicio del alineamiento.

El algoritmo SW tiene complejidad temporal cuadrática. El puntaje de alineamiento óptimo se puede calcular linealmente en cuanto a espacio. Sin embargo, para obtener el alineamiento óptimo entre las secuencias resulta necesario almacenar la matriz H completamente, por lo que su complejidad espacial se vuelve cuadrática.

Resulta importante notar que ninguna celda de la matriz puede tener valor inferior a 0. Adicionalmente, existe un orden estricto en el cómputo de la matriz H debido a las dependencias de datos inherentes al problema. Toda celda en la matriz H tiene dependencias sobre otras tres celdas: la que está a la izquierda, la que está por encima y la anterior en la diagonal.

2.2. Aceleración del algoritmo SW

Esta sección se enfoca en el estado del arte de la aceleración del algoritmo SW.

2.2.1. Dependencias de datos y paralelismo

Aunque las dependencias mencionadas en la Sección 2.1 restringen las formas en que se puede computar la matriz de similitud, el algoritmo SW posee cierto paralelismo inherente que puede ser explotado para reducir su costo computacional. En general, las implementaciones suelen emplear alguno de los dos siguiente es esquemas:

- El esquema intra-tarea se basa en acelerar el cómputo de un único alineamiento por vez. Las implementaciones que adoptan este esquema suelen computar las celdas de cada antidiagonal de la matriz en paralelo, aprovechando que estos cálculos son independientes entre sí. También es posible computar varias celdas de una _la o de una columna simultáneamente; sin embargo, como este enfoque ignora las dependencias de datos antes mencionadas, se requiere de un ajuste posterior de los valores de las celdas para mantener la corrección del algoritmo.
- El esquema inter-tarea se basa en acelerar el cómputo de múltiples alineamientos al mismo tiempo. La mayor ventaja de este enfoque es la independencia en el cálculo de cada alineamiento.

Ambos enfoques han sido explorados por la comunidad científica en una amplia variedad de trabajos y dispositivos, describiéndose los más destacados en las siguientes secciones.

2.2.2. Implementaciones existentes

Para poder evaluar el desempeño de una implementación en forma independiente de las secuencias de consulta y de las bases de datos utilizadas para las diferentes pruebas al igual que de la arquitectura de soporte, se emplea la métrica de rendimiento Cell Updates Per Second (CUPS). Dada una secuencia de consulta Q y una base de datos D, el valor de GCUPS (mil millones de CUPS) se calcula como jQj_jDj t_109 , donde jQj es el número de total de residuos en la secuencia de consulta, jDj es el número total de residuos en la base de datos y t es el tiempo de ejecución en segundos requerido para computar las matrices de similitud [13]. A continuación, se describen las implementaciones más destacadas de acuerdo a la arquitectura de soporte empleada.

Implementaciones en CPU. Las implementaciones en CPU se han valido de instrucciones vectoriales para acelerar el cómputo de las matrices de similitud. En general, se pueden identificar diferentes enfoques para la utilización de vectorización en el cómputo de las matrices de similitud, los cuales se ilustran en la Figura 1. La mayoría de los esfuerzos se han concentrado en el enfoque intra-tarea. Sin embargo, es el enfoque inter-tarea el que ha producido los mejores resultados: en el año 2011, Rognes presentó SWIPE, una implementación paralela que explota las instrucciones SSSE3 bajo el enfoque inter-tarea [14]. SWIPE mostró rendimientos superiores a sus implementaciones contemporáneas por lo que se la considera la implementación SW más rápida para el conjunto de instrucciones SSE.

Implementaciones en GPU. Entre las propuestas basadas en GPU, se destaca CUDASW++ [13] y sus versiones posteriores [15, 16]. En su última versión, CUDASW++ combina cómputo concurrente en la CPU y en la GPU empleando el esquema inter-tarea en ambos casos. También se caracteriza por una cuidadosa organización de los accesos de memoria para aprovechar acceso coalescente y por la explotación de memoria

constante para las secuencias de consulta y de memoria compartida para la matriz de sustitución. En la actualidad, se la considera la implementación SW más rápida para sistemas basados en GPUs compatibles con CUDA.
Implementaciones en FPGA. En general, las implementaciones en FPGA se basan en crear bloques básicos que sean capaces de computar el valor de una celda de una matriz en cada ciclo de reloj. Luego, múltiples instancias de estos bloques son combinados a la vez para formar arreglos sistólicos que puedan procesar grandes cantidades de datos en paralelo. Desafortunadamente se hace difícil realizar una comparación analítica que resulte justa debido a diferentes causas [17, 18]: (1) existe una amplia variedad en tipos de FPGAs y cada una implementa sus circuitos de forma diferente, lo que dificulta una comparación directa; (2) existen muy pocas implementaciones que sean completas desde el punto de vista funcional, mientras que las que reportan los mejores resultados suelen contemplar pruebas sintéticas con el objetivo de mostrar la viabilidad del empleo de esta clase de aceleradores aunque su uso potencial en el mundo real es limitado; y (3) falta de documentación en los detalles de implementación que resultan claves para la comparación.

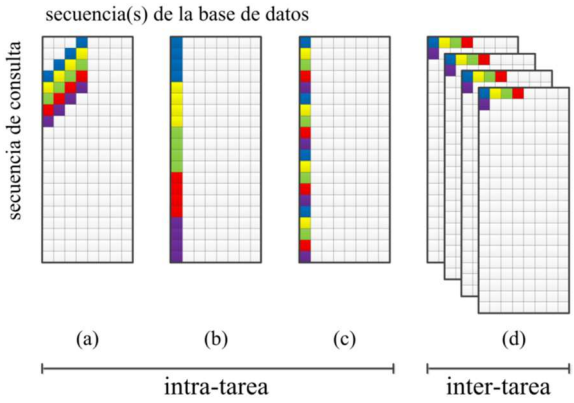

Figura 1: Enfoques para paralelización SIMD del algoritmo Smith-Waterman (adaptado de [14]). (a) Vectorización en las anti-diagonales. (b) Vectorización en la secuencia de consulta. (c) Vectorización por franjas en la secuencia de consulta. (d) Vectorización en múltiples secuencias de la base de datos.

Implementaciones en Xeon Phi. En el año 2014, Liu y Schmidt presentaron la herramienta SWAPHI [19] basada en el modelo o_oad de OpenMP. SWAPHI soporta los esquemas de paralelismo intra-tarea e inter-tarea, además de diferentes técnicas para la obtención de los puntajes de sustitución. XSW [20] es una herramienta alternativa publicada en el año 2014. Al igual que SWAPHI, XSW emplea el esquema inter-tarea. Sin embargo, a diferencia de SWAPHI, XSW se ejecuta en modo nativo con Pthreads como modelo de programación. Una versión extendida de XSW, conocida como XSW 2.0, fue desarrollada posteriormente [21]. Esta implementación adopta el modelo o_oad para combinar cómputo concurrente en la CPU.

3. Contribuciones y resultados

A continuación se describen las tres principales contribuciones de esta tesis, las cuales se encuentran sustentadas por los siguientes trabajos publicados en congresos internacionales [22, 23], revistas indexadas [24, 25] y capítulos de libros [26].

Optimización de SW para sistemas heterogéneos basados en aceleradores Xeon Phi: SWIMM

La primera contribución de esta tesis consiste en la introducción de la herramienta SWIMM para búsquedas de similitud[1] en sistemas heterogéneos basados en aceleradores Xeon Phi, la cual trabaja con bases de datos biológicas reales y permite configurar los parámetros de ejecución. Además, admite tres modos de ejecución: (1) cómputo en el host, (2) cómputo en los coprocesadores y (3) cómputo concurrente en el host y los coprocesadores. Mediante esta herramienta, se estudió y evaluó la viabilidad de esta clase de sistemas heterogéneos para el alineamiento de secuencias de proteínas mediante el algoritmo SW. Para beneficio de la comunidad científica, SWIMM se encuentra disponible en un repositorio web público[2].

En primer lugar, considerando la compatibilidad de código entre los procesadores Xeon y los coproces adores Xeon Phi, se desarrolló una implementación común para ambos dispositivos siguiendo el enfoque inter-tarea. La propuesta explota dos niveles de paralelismo: a nivel de hilo mediante OpenMP y a nivel de datos usando vectorización guiada. Luego, se desarrollaron versiones alternativas para cada dispositivo que reemplazan la vectorización guiada por vectorización manual a través de funciones intrínsecas. En todos los casos se estudiaron y aplicaron diferentes técnicas de programación y optimización a las distintas implementaciones. Finalmente, utilizando los códigos desarrollados para ambos dispositivos se implementó una versión híbrida capaz de computar alineamientos en el Xeon y en el Xeon Phi de forma simultánea. Esta implementación admite dos modos de distribución de la carga de trabajo (estática y dinámica).

Posteriormente, se seleccionaron secuencias de consulta y bases de datos de proteínas representativas para la comunidad científica que sirvieran como benchmarks. En particular, se eligieron 20 secuencias de consulta de variada longitud y las bases de datos de datos Swiss-Prot y Environmental NR, las cuales fueron utilizadas conjuntamente en investigaciones similares anteriormente [14, 16, 19, 20]. Además, se seleccionaron otras implementaciones de referencia con las cuales poder realizar comparaciones: SWIPE para CPU, SWAPHI para Xeon Phi, XSW 2.0 para CPU-Xeon Phi y CUDASW++ para CPU-GPU. =

Se probaron las distintas implementaciones sobre dos arquitecturas heterogéneas diferentes. A continuación se detallan los principales resultados y conclusiones del análisis experimental:

- Para lograr implementaciones SW de alto rendimiento tanto en los Xeon como en los Xeon Phi, se adaptó el código con el _n de:
  - Explotar dos niveles de paralelismo: usar más hilos no resulta suficiente; se debe explotar paralelismo de datos también. En ese sentido, la vectorización guiada ofrece portabilidad de código y un menor costo de programación. Sin embargo, la mejora de rendimiento que se puede lograr es pequeña comparada a la de la vectorización manual.
  - Aprovechar el hecho de que los puntajes de similitud no requieren de un amplio rango de representación. Mediante la explotación de tipos de dato enteros de bajo rango es posible aumentar el paralelismo a nivel de datos (más elementos por vector). En particular, se mostró que resulta conveniente computar los alineamientos usando enteros de 8 bits en primera instancia y utilizar progresivamente representaciones más amplias cuando sea necesario.
  - Emplear técnicas de procesamientos por bloques que mejoren la localidad de datos, especialmente en el Xeon Phi considerando que su memoria caché es más pequeña.
  - Lograr una carga de trabajo balanceada entre el Xeon y el Xeon Phi en versiones híbridas. Una diferencia de potencia computacional (en GCUPS) muy grande entre ambos dispositivos puede llevar a un serio desbalance de carga, lo que se traslada a una reducción del rendimiento.
- Con respecto a los GCUPS, la implementación con intrínsecas SSE logró 118 GCUPS en el Xeon E5-2670 con 32 hilos mientras que su equivalente con AVX2 alcanzó 355 GCUPS en el Xeon E5-2695 v3 con 56 hilos. Aun así, la implementación con KNC sólo obtuvo un máximo de 50 GCUPS para 228 hilos. El pobre desempeño del Xeon Phi se debe a la ausencia de capacidades vectoriales para explotar enteros de menos de 32 bit.
- En cuanto a la comparación con otras implementaciones SW, la versión SSE de SWIMM se mostró equiparable con SWIPE mientras que la versión AVX2 logró superarlo ampliamente alcanzado diferencias de hasta 1.4_. Cabe destacar que, hasta donde llega el conocimiento del tesista, esta es la primera implementación SW para el conjunto de instrucciones AVX2. Por otra parte, la versión KNC de SWIMM se mostró competitiva con SWAPHI, siendo capaz de superarlo para secuencias medianas y largas. Complementariamente, la versión híbrida de SWIMM con distribución dinámica (SSE+KNC) superó notablemente a su alternativa XSW 2.0, con mayores diferencias para secuencias de consulta largas. Por último, en el sistema basado en Xeon E5-2695 v3, SWIMM (versión AVX2+KNC) logró mejores rendimientos que CUDASW++ 3.0 utilizando los recursos del host y una GPU NVIDIA K20c.
  Resulta importante mencionar que la próxima generación de coprocesadores Xeon Phi (Knights Landing) unificará su conjunto de instrucciones con la generación actual de procesadores Xeon (Skylake) adoptando las extensiones AVX-512 de 512 bits. Como este conjunto de instrucciones sí posee capacidades para explotar enteros de bajo rango, se espera un incremento notable en el rendimiento de ambos dispositivos usando la metodología propuesta.

Optimización de SW para sistemas heterogéneos basados en aceleradores FPGA: OSWALD

La segunda contribución de esta tesis consiste en la presentación de la herramienta OSWALD para búsquedas biológicas en sistemas heterogéneos basados en aceleradores FPGA, la cual trabaja con bases de datos biológicas reales y permite configurar los parámetros de ejecución. Mediante esta herramienta, se estudió y evaluó la viabilidad de esta clase de sistemas heterogéneos para el alineamiento de secuencias de proteínas mediante el algoritmo SW. A diferencia de las implementaciones disponibles en la literatura, se decidió explorar los beneficios de utilizar una tecnología innovadora, como lo es OpenCL en el ámbito de las FPGAs, en lugar de HDLs tradicionales como VHDL o Verilog. Al igual que la herramienta anterior, OSWALD se encuentra disponible en un repositorio web público para beneficio de la comunidad científica[3].

Como primer paso, se desarrolló una implementación para una única FPGA. En cuanto a la paralelización de los alineamientos se optó por el esquema inter-tarea. Para el desarrollo del algoritmo, se estudiaron y aplicaron diferentes técnicas de programación y optimización. Luego, se evaluó el rendimiento y el consumo de recursos de las diferentes versiones desarrolladas de forma de encontrar la configuración más beneficiosa. A continuación, se extendió la implementación mono-FPGA para que sea capaz de soportar ejecución en múltiples aceleradores al mismo tiempo. Finalmente, utilizando el código Xeon de SWIMM se implementó una versión híbrida capaz de computar alineamientos en la CPU y en las FPGAs de forma simultánea.

Empleando las secuencias de consulta y las bases de datos de proteínas utilizadas anteriormente, se probó la implementación multi-FPGA y la implementación híbrida sobre dos arquitecturas heterogéneas diferentes. También se probaron otras implementaciones de referencia con las cuales comparar las propias: SWIMM para sistemas heterogéneos basados en Xeon Phi y CUDASW++ 3.0 para los basados en GPUs compatibles con CUDA. Lamentablemente no fue posible realizar una comparación con otras implementaciones FPGA debido a la falta de disponibilidad del código fuente. Si bien un análisis teórico es posible, las significativas diferencias entre las distintas implementaciones impiden una comparación justa. A continuación se detallan los principales resultados y conclusiones del análisis experimental:

- De la exploración de los diferentes kernels SW se puede extraer:
  - Al computar las matrices de alineamiento empleando técnicas de procesamientos por bloques no sólo se mejora la localidad de datos sino que también se reduce los requerimientos de memoria del kernel, lo que favorece el uso de memoria privada de baja latencia.
  - Al explotar un mayor grado de paralelismo de datos a través de la vectorización de operaciones se logra incrementar el rendimiento en forma notable a expensas de un aumento moderado en el consumo de recursos.
  - Al explotar tipos de dato enteros de bajo rango no sólo se reduce el consumo de recursos sino que también se mejora el rendimiento. De esta forma se aprovecha el hecho de que los puntajes de similitud no requieren de un amplio rango de representación. En particular, se mostró que resulta conveniente computar los alineamientos usando enteros de 8 bits en la FPGA y recomputar en el host cuando sea necesario.
  - Lograr una carga balanceada entre los diferentes dispositivos de procesamientos
  - es un factor clave para obtener alto rendimiento. En la implementación multi-FPGA, ésto se logra en la etapa de preprocesamiento al dividir la base de datos en número de chunks que sea múltiplo del número de FPGAs y procurando que los chunks contengan aproximadamente la misma cantidad de residuos. Por su parte, la implementación híbrida requiere de un esquema más complejo por las diferentes capacidades computacionales que pueden tener el host y los aceleradores. Al estimar la potencia de cómputo de cada dispositivo de procesamiento, no sólo se logra una carga de trabajo balanceada con un overhead mínimo sino también una implementación general para cualquier sistema heterogéneo de esta clase.
- Con respecto a los GCUPS, la implementación mono-FPGA logró 58.4 GCUPS mientras que la implementación multi-FPGA con dos aceleradores obtuvo 114 GCUPS, lo que representa una escalabilidad casi lineal. Por su parte, la implementación híbrida alcanzó 178.9 y 401.1 GCUPS en los sistemas basados basados en Xeon E5-2670 y en Xeon E5-2695 v3, respectivamente.

[3] *OSWALD se encuentra disponible online en https://github.com/enzorucci/OSWALD*

- En cuanto a la comparación con otras implementaciones SW, OSWALD y SWIMM mostraron comportamientos similares en los dos sistemas heterogéneos utilizados. El rendimiento de OSWALD estuvo siempre por encima del de SWIMM (incluso para secuencias cortas), logrando mayores diferencias a medida que la carga de trabajo aumentaba. Por su parte, CUDASW++ 3.0 superó a todas las demás implementaciones en el sistema basado en Xeon E5-2670, principalmente por el gran poder computacional de la GPU NVIDIA K20c. En contraste, presentó el peor rendimiento en el sistema basado en Xeon E5-2695 v3 debido a su incapacidad de aprovechar las extensiones AVX2.

El costo de programación y la falta de portabilidad en los códigos FPGA han limitado tradicionalmente su aplicación a las búsquedas SW. OSWALD es una implementación portable, completamente funcional y general para acelerar búsquedas de similitud en sistemas heterogéneos basados en FPGA que demostró ser competitivo con otras implementaciones SW de referencia.

Eficiencia energética de SW en sistemas heterogéneos

Como en la actualidad la comunidad HPC no sólo está interesada en mejorar el rendimiento sino también en reducir el consumo energético, la tercera contribución de esta tesis consiste en el estudio y evaluación de la eficiencia energética de diferentes sistemas heterogéneos para el alineamiento de secuencias de proteínas mediante el algoritmo SW. Para ello, no se realizaron pruebas especiales sino que se monitorizó el consumo de potencia por parte de los diferentes sistemas en los experimentos realizados anteriormente.

En primer lugar, se analizó el consumo de potencia de las implementaciones híbridas de SWIMM y de OSWALD en forma individual. Posteriormente, se analizó comparativamente el rendimiento y el consumo de potencia obtenidos en las diferentes arquitecturas heterogéneas utilizadas. A continuación se detallan los principales resultados y conclusiones del análisis experimental:

- Algunas de las técnicas que, además de incrementar los GCUPS, mejoraron la eficiencia energética son:
  - Explotar mayor paralelismo de datos enteros de bajo rango, como se evidenció en el sistema basado en Xeon E5-2695 v3 (AVX2) comparado al basado en Xeon E5-2670 (SSE).
  - Usar Hyper-Threading. El uso de esta tecnología provocó un incremento mayor en el rendimiento que el correspondiente al consumo de potencia en ambas arquitecturas.
  - Lograr una carga de trabajo equilibrada entre host y aceleradores. De esta manera, no sólo se mejoraron las prestaciones sino que también se redujo el consumo energético.
- En cuanto a la computación basada únicamente en aceleradores (sin ejecución simultánea del host):
  - La FPGA puede no ser la mejor opción desde el punto de vista del rendimiento. Sin embargo, sí lo puede ser desde la perspectiva del consumo de potencia; su bajo consumo puede ser de utilidad cuando éste aspecto sea la mayor prioridad.
  - El Xeon Phi no representa una buena elección tanto desde la óptica del rendimiento como del consumo de potencia.
- A través de SWIMM, las CPUs presentaron un buen balance entre rendimiento y consumo. Sus prestaciones mejoraron significativamente al explotar las extensiones AVX2. Respecto a la combinación con aceleradores:
  - El uso de arquitecturas heterogéneas basadas en coprocesadores Xeon Phi no fue una buena opción en términos del consumo de potencia. La integración del Xeon Phi al cómputo de los alineamientos proveyó poca mejora de rendimiento al host y decrementó su cociente GCUPS/Watt. Se espera que la incorporación futura de las extensiones AVX-512 cambie esta situación.
  - En el sistema basado en Xeon E5-2670, CUDASW++ 3.0 mejoró los GCUPS y el cociente GCUPS/Watt de SWIMM (versión Xeon). Sin embargo, no fue capaz de repetir sus prestaciones en el sistema basado en Xeon E5-2695 v3 debido a que CUDASW++ 3.0 sólo explota las instrucciones SSE. Por ese motivo, la combinación de CPUs con GPUs puede ser una buena alternativa en sistemas que puedan explotar las instrucciones SSE, pero que no dispongan de las extensiones AVX2. El desarrollo de una herramienta que dé soporte a las extensiones AVX2 permitiría aumentar la eficiencia energética de esta clase de sistemas.
  - La computación híbrida basada en CPU-FPGA se ubicó como la opción más eficiente energéticamenteteniendo en cuenta que OSWALD obtuvo los cocientes GCUPS/Watt más altos en

ambos sistemas. Aun más, la FPGA fue el único acelerador que mejoró las tasas GCUPS/Watt del host en ambos sistemas.

Aunque existen otras evaluaciones de eficiencia energética en el contexto de SW [27, 28], éstas poseen uno o más limitaciones que reducen su utilidad en el mundo real. En esta evaluación, se han empleado secuencias de consulta y bases de datos representativas de la comunidad bioinformática, potentes arquitecturas orientadas a HPC y eficientes implementaciones que han demostrado ser las más rápidas de su clase. En base a estas condiciones y teniendo en cuenta el rol de SW en la bioinformática, se espera que esta evaluación le resulte útil a cualquier centro de investigación y desarrollo de este área al momento de seleccionar un sistema HPC para sus aplicaciones de acuerdo con las prioridades que posea.

4. Conclusiones

De forma de procesar el creciente volumen de información biológica con tiempos de respuesta aceptables, resulta necesario desarrollar nuevas herramientas computacionales que sean capaces de acelerar primitivas claves y algoritmos elementales eficientemente en términos de rendimiento y consumo energético. Por ese motivo, esta tesis se ha planteado como objetivo general evaluar el rendimiento y la eficiencia energética de sistemas para cómputo de altas prestaciones al acelerar el alineamiento de secuencias biológicas mediante el método de Smith-Waterman.

En primer lugar, se estudiaron las posibles formas de paralelizar el algoritmo SW y se describieron las implementaciones existentes sobre diferentes plataformas de procesamiento: CPU, GPU, FPGA y Xeon Phi. El análisis para cada dispositivo incluyó la evolución temporal de sus implementaciones así como también una descripción detallada de los aportes, las limitaciones, los resultados y la experimentación realizada en cada uno de los trabajos. Este análisis fue posible gracias al estudio de las diferentes arquitecturas y modelos de programación además del de los distintos algoritmos para alineamiento de secuencias biológicas llevado al inicio de la tesis.

El siguiente paso consistió en desarrollar nuevas soluciones algorítmicas para sistemas heterogéneos. Como se mencionó en la Sección 1.3, se decidió priorizar el desarrollo de nuevas soluciones algorítmicas para sistemas heterogéneos basados en Xeon Phi y basados en FPGA. Como punto de partida, se desarrollaron y optimizaron implementaciones para las CPUs y los Xeon Phi en forma individual antes de combinarlos en una implementación híbrida. La integración de estas implementaciones dio como resultado la herramienta SWIMM. A partir del análisis inicial del estado del arte, se identificó a SWIPE como la herramienta más rápida para búsquedas de similitud en CPU. Mediante los experimentos realizados, se mostró que la versión SSE de SWIMM es equiparable con SWIPE mientras que la versión AVX2 logró superarlo ampliamente alcanzado diferencias de hasta 1.4_. Cabe destacar que, hasta donde llega el conocimiento del tesista, esta es la primera implementación SW para el conjunto de instrucciones AVX2. En cuanto a los Xeon Phi, al inicio de esta investigación no existían implementaciones disponibles para este coprocesador. Sin embargo, en el año 2014 aparecieron SWAPHI y XSW 2.0. Mientras que SWAPHI sólo explota el coprocesador, XSW 2.0 es capaz de aprovechar la potencia del host en simultáneo. A través de los experimentos realizados, se mostró que la versión KNC de SWIMM resulta competitiva con SWAPHI, siendo capaz de superarlo para secuencias medianas y largas. Complementariamente, la versión híbrida de SWIMM con distribución dinámica (SSE+KNC) superó notablemente a su alternativa XSW 2.0, logrando aceleraciones de hasta 3.9_.

El siguiente paso en el trabajo experimental fue el desarrollo de soluciones algorítmicas para sistemas heterogéneos basados en aceleradores FPGA. A diferencia de las implementaciones disponibles en la literatura, se decidió explorar los beneficios de utilizar una tecnología innovadora, como lo es OpenCL en el ámbito de las FPGAs. En primer lugar, se exploró el rendimiento y el consumo de recursos de diferentes kernels de forma de encontrar la configuración más beneficiosa. Posteriormente, se desarrolló una versión híbrida empleando el código de SWIMM,siendo compilada junto a las anteriores en la herramienta OSWALD. Hasta donde llega el conocimiento del tesista, ésta la primera implementación utilizando OpenCL sobre FPGAs para búsquedas de similitud. A demás, en base al análisis inicial del estado del arte, es una de las pocas implementaciones que es completamente funcional y general para esta clase de sistemas, además de facilitar la portabilidad por el empleo de OpenCL. Desafortunadamente, la ausencia del código fuente impide una comparación con otras implementaciones basadas en FPGA y aunque un análisis teórico es posible, los diferentes dispositivos, tecnologías y enfoques empleados no sólo complican una comparación directa sino que también dificultan hacerla en forma justa.

El último paso del trabajo experimental consistió en relacionar el rendimiento alcanzado por los diferentes sistemas empleados con el correspondiente consumo de potencia. De acuerdo al análisis realizado, se puede afirmar que los sistemas basados únicamente en CPU son capaces de obtener un buen balance entre rendimiento y consumo de potencia a partir de la explotación de multihilado e instrucciones vectoriales, destacándose aquellos que disponen de las extensiones AVX2. Por otra parte, la incorporación de aceleradores al cómputo del host demostró mejorar el rendimiento global del sistema en todos los casos, aunque la proporción de mejora varió de acuerdo al acelerador elegido. Desafortunadamente no ocurrió lo mismo en cuanto a la e_ciencia energética. Los Xeon Phi no representan una buena opción para este tipo de aplicación. La ausencia de capacidades vectoriales para enteros de bajo rango es la causa principal del pobre rendimiento de este coprocesador, lo que provoca que el incremento en el consumo energético sea mucho mayor que la ganancia de rendimiento. En sentido opuesto, las GPUs sí pueden explotar este tipo de datos y es por ello que se puede aprovechar su gran poder computacional para acelerar los alineamientos. Aunque el aumento en el consumo de potencia es elevado, la mejora lograda por el uso de esta clase de aceleradores resulta superior, lo que se traduce en una mayor eficiencia energética. En el caso de las FPGAs, su capacidad de reconfiguración hace posible adaptar el hardware a los requerimientos del algoritmo SW. Si bien el rendimiento alcanzable de estos aceleradores puede ser inferior al correspondiente a las GPUs, su bajo consumo de potencia lleva a mayores tasas de eficiencia energética. Por último, a diferencia de otras evaluaciones de rendimiento y eficiencia energética en el contexto de SW, en esta tesis se han empleado secuencias de consulta y bases de datos representativas de la comunidad bioinformática, potentes arquitecturas orientadas a HPC y eficientes implementaciones que han demostrado ser las más rápidas de su clase. Por ese motivo se considera que la evaluación presentada puede ser de mayor utilidad en el mundo real. De acuerdo a los resultados obtenidos y a las contribuciones realizadas, se espera que esta tesis aporte a una mayor adopción de SW por parte de la comunidad bioinformática y a un procesamiento más eficiente de los alineamientos de secuencias biológicas en términos de rendimiento y consumo energético.

5. Líneas de trabajo futuro

Algunas de las líneas de trabajo futuro que se desprenden de esta tesis pueden ser:

- Desarrollar soluciones algorítmicas para las nuevas generaciones de procesadores Xeon y Xeon Phi. Como se mencionó anteriormente, Intel ha anunciado que unificará el conjunto de instrucciones en las próximas generaciones de los Xeon y los Xeon Phi al integrar las extensiones AVX-512 en ambos dispositivos. Este conjunto de instrucciones dispone de capacidades para explotar enteros de bajo rango, lo que representa una de los principales factores para obtener alto rendimiento. Teniendo en cuenta los resultados obtenidos con la metodología empleada, se espera un incremento notable en el rendimiento de ambos dispositivos.
- Explorar la explotación de dispositivos no convencionales en HPC para la aceleración del alineamiento de secuencias, como pueden ser los DSPs o las arquitecturas basadas en procesadores ARM. Si bien estos dispositivos poseen capacidades computacionales limitadas, su bajo consumo energético los vuelve atractivos desde el punto de vista de la eficiencia energética.
- Extender la evaluación de rendimiento y eficiencia energética a otros dominios. Así como el algoritmo SW resulta representativo de la bioinformática, interesa extender el análisis realizado a aplicaciones que resulten características de otras áreas.

Referencias

[1] W. Feng, X. Feng, and R. Ge, _Green supercomputing comes of age,_ IT Professional, vol. 10, no. 1, pp. 17_23, 2008.
[2] Top500. [Online]. Available: www.top5000.org
[3] Green500. [Online]. Available: www.green500.org
[4] _Front matter,_ in High Performance Parallelism Pearls, J. R. Je_ers, Ed. Boston: M. Kaufmann, 2015, pp. i_ii.
[5] (2015, 06) Intel to acquire altera. HPC Wire. [Online]. Available: http://www.hpcwire.com/o_-the-wire/intel-to-acquire-altera/
[6] M. Vestias and H. Neto, _Trends of CPU, GPU and FPGA for high-performance computing,_ in Field Programmable Logic and Applications (FPL), 2014 24th International Conference on, Sept 2014, pp. 6.
[7] S. O. Settle, _High-performance Dynamic Programming on FPGAs with OpenCL, 2014.
[8] National Center for Biotechnology Information. [Online]. Available: http://www.ncbi.nlm.nih.gov/
[9] B. Schmidt, Bioinformatics: High Performance Parallel Computer Architectures, B. Schmidt, Ed. CRC Press, 2010.

[10] T. F. Smith and M. S. Waterman, _Identi_cation of common molecular subsequences,_ Journal of Molecular Biology, vol. 147, no. 1, pp. 195_197, March 1981.
[11] A. Siegel. (2011, July) What are the advantages/disadvantages of using Protein Alignment vs DNA Alignment? [Online]. Available: https://www.ocf.berkeley.edu/~asiegel/posts/?p=14
[12] O. Gotoh, _An improved algorithm for matching biological sequences,_ in J. of Mol. Biol., vol. 162, 1981, pp. 705_708.
[13] Y. Liu, D. L. Maskell, and B. Schmidt, _CUDASW++: optimizing Smith-Waterman sequence database searches for CUDA-enabled graphics processing units,_ BMC Research Notes, vol. 2:73, 2009.
[14] T. Rognes, _Faster Smith-Waterman database searches with inter-sequence SIMD parallelization,_ BMC Bioinformatics, vol. 12:221, 2011.
[15] Y. Liu, B. Schmidt, and D. Maskell, _CUDASW++2.0: enhanced Smith-Waterman protein database search on CUDAenabled
GPUs based on SIMT and virtualized SIMD abstractions,_ BMC Research Notes, vol. 3, no. 1, p. 93, 2010.
[16] Y. Liu, A. Wirawan, and B. Schmidt, _CUDASW++ 3.0: accelerating Smith-Waterman protein database search by coupling CPU and GPU SIMD instructions,_ vol. 14:117, 2013.
[17] L. Hasan and Z. Al-Ars, Computational Biology and Applied Bioinformatics. InTech, 2011, ch. 9, pp. 187_202.
[18] S. Dydel and P. Bala, _Large Scale Protein Sequence Alignment Using FPGA Reprogrammable Logic Devices,_ in Field Programmable Logic and Application, ser. LNCS. Springer, 2004, vol. 3203, pp. 23_32.
[19] Y. Liu and B. Schmidt, _SWAPHI: Smith-Waterman protein database search on Xeon Phi coprocessors,_ in 25th IEEE International Conference on Application-speci_c Systems, Architectures and Processors (ASAP 2014), 2014.
[20] L. Wang, Y. Chan, X. Duan, H. Lan, X. Meng, and W. Liu, _XSW: Accelerating Biological Database Search on Xeon Phi,_ in Fourth International Workshop on Accelerators and Hybrid Exascale Systems 2014, 2014.
[21] __. (2014) XSW 2.0: A fast Smith-Waterman Algorithm Implementation on Intel Xeon Phi Coprocessors. [Online]. Available: http://sdu-hpcl.github.io/XSW/
[22] E. Rucci, A. De Giusti, M. Naiouf, G. Botella, C. Garcia, and M. Prieto-Matias, _Smith-Waterman algorithm on heterogeneous systems: A case study,_ in 2014 IEEE International Conference on Cluster Computing (CLUSTER), Sep. 2014, pp. 323_330.
[23] E. Rucci, C. García, G. Botella, A. De Giusti, M. Naiouf, and M. Prieto-Matias, _Smith-Waterman Protein Search with OpenCL on an FPGA,_ in 2015 IEEE Trustcom/BigDataSE/ISPA, vol. 3, Aug. 2015, pp. 208_213.
[24] __, _An energy-aware performance analysis of SWIMM: Smith-Waterman implementation on Intel's Multicore and Manycore architectures,_ Concurrency and Computation: Practice and Experience, vol. 27, no. 18, pp. 5517_5537, 2015.
[25] __, _OSWALD OpenCL Smith-Waterman on Altera's FPGA for Large Protein Databases,_ International Journal of High Performance Computing Applications, Jun. 2016.
[26] __, _State-of-the-Art in Smith-Waterman Protein Database Search on HPC Platforms,_ in Big Data Analytics in Genomics, K.-C. Wong, Ed. Springer, Oct. 2016, pp. 197_223, dOI: 10.1007/978-3-319-41279-5_6.
[27] K. Benkrid, A. Akoglu, C. Ling, Y. Song, Y. Liu, and X. Tian, _High Performance Biological Pairwise Sequence Alignment: FPGA Versus GPU Versus Cell BE Versus GPP,_ Int. J. Recon_g. Comput., vol. 2012, pp. 7:7_7:7, 2012.
[28] D. Zou, Y. Dou, and F. Xia, _Optimization schemes and performance evaluation of Smith-Waterman algorithm on CPU, GPU and FPGA,_ Concurr. Comput., vol. 24, no. 14, pp. 1625_1644, 2012.